UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/670,836	09/25/2003	Maximino Aguilar JR.	AUS920030791US1	8258

40412 7590 10/09/2007

IBM CORPORATION- AUSTIN (JVL)
C/O VAN LEEUWEN & VAN LEEUWEN
PO BOX 90609
AUSTIN, TX 78709-0609

EXAMINER	
INGBERG, TODD D	

ART UNIT	PAPER NUMBER
2193	

MAIL DATE	DELIVERY MODE
10/09/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/670,836	AGUILAR ET AL.	
	Examiner	Art Unit	
	Todd Ingberg	2193	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *16 July 2007*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-30* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-30* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *7/9/07* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *7/16/07,7/9/07,5/20/07*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Claims 1 – 30 have been examined.

Claims 1, 4, 5, 7, 9, 11, 14, 15, 17, 19, 21-27, 29 – 30 have been amended.

Drawings

1. The drawings resubmitted on July 9, 2007 have been entered. Examiner Thanks the Applicant for correcting the PTO scanning problem.

Information Disclosure Statement

2. The Information Disclosure Statements filed July 16, 2007, July 9, 2007 and May 20, 2007 have been considered.

Claim Rejections - 35 USC § 112

3. The rejection under second paragraph of 35 U.S.C. 112 has been overcome.

Terms and Examiner Interpretation

4. Applicant's terms are well supported. The Examiner is making of record terms of the art which are related to the computing environment of the claimed invention.

Tightly coupled – "... two or more processing units share real storage are controlled by the same control program and can communicate directly with each other." IBM Dictionary

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. Claims 1 – 30 are rejected under 35 U.S.C. 103(a) as being unpatentable over USPN# 5490,278 Mochizuki (Moch) and USPN # 5,887,186 Nakanishi (Nak) in view of USPN # 7,165,108 Matena et al. filed March 19, 2001.

Moch and Nak teach tightly coupled heterogeneous processors and processing requests between the two processors and the interactions and processing between the processors. But Neither Moch and Nak teach a load balancing with virtual machine program. It Matena who teaches load balancing

Claim 1

Moch and Nak teach a computer-implemented method for processing software code (Moch, Abstract, processing with back-substitution), said method comprising:
loading, at a second processor, a virtual machine program into a local memory corresponding to the second processor (Matena, Abstract); receiving, at the second processor, a code processing request requested by a first processor, (Moch, Abstract, multiprocessor) wherein the first and second processors are heterogeneous processors within a computer system that share a common memory (Moch, Figure 1, #1 - Abstract, multiprocessor – tightly coupled);**reading, from the common memory shared by the first and second processors, software code data corresponding to the request, the software code dataincluding virtual machine code adapted to be processed by the virtual machine program (Matena, Abstract);** writing **the** software code data corresponding to the request to the local memory corresponding to the second processor in response to the request; processing the software code data by the second processor (Nak, Abstract, Rearrage and transfer), **wherein the processing includes processing the virtual machine code at the second processor using the virtual machine program, the processing resulting in executable instructions; writing the executable instructions to a memory location accessible by the first processor; and executing, at the first processor, the executable instructions (Matena, Abstract – JAVA environment and load balancing between the two processors);** Moch teaches a multiprocessor environment and Nak teaches rearranging the memory and transferring data. Therefore, it would have been obvious to one of ordinary skill in the art at the time of invention to use the combination of Moch and Nak with the teaching of the ability to perform substitution process in parallel in each process with the Load balancing of Matena because optimization techniques make computing more efficient.

Claim 2

The method as described in claim 1 further comprising: prior to the receiving: reading script code from the common memory; writing the script code to a local memory corresponding to the first processor; interpreting, at the first processor, the script code, the

interpreting resulting in the software code; and writing the software code to the second processor's local memory. As per claim (Nak).

Claim 3
The method as described in claim 1 further comprising: writing data resulting from the executing to the common memory. As per claim 1.

Claim 4
The method as described in claim 1 further comprising: prior to the receiving:
running a first program, **in response to running the first program, identifying a call to a software effect corresponding to the software code data (What to balance in Matena, Abstract); an** loading the software code data into the common memory, wherein the processing of the software code is data occurs during the running of the first program and wherein the processing is completed prior to the program calling the software effect. As per claim 1 and Figure 23, #243.

(Note to Applicant – Bolding and underling of amendments stops here.)
Claim 5
The method as described in claim 4 further comprising: performing by the second processor, a multimedia effect resulting from the processing of the software code data.
As per claim 1.

Claim 6
The method as described in claim 4 further comprising: receiving, at the first processor, the executable instructions resulting from the processing performed by the second processor, wherein the executable instructions are adapted to perform a multimedia effect; and performing the multimedia effect on the first processor by executing the received executable instructions.
As per claim 1.

Claim 7
The method as described in claim 1 wherein the writing further comprising: writing the executable instructions to a memory location accessible by the first processor using a direct memory access (DMA) operation. As per claim 1.

Claim 8
The method as described in claim 7 wherein the memory location is selected from the group consisting of a local memory corresponding to the first processor, and the common memory.
As per claim 1.

Claim 9
The method as described in claim 1 wherein the first processor has a first instruction set architecture and the second processor has a second instruction set architecture (heterogeneous by meaning) and wherein the executable instructions resulting the processing performed by the

second processor are adapted to be executed on the first processor and not the second processor (emulation inherently part of the heterogeneous environment) As per claim 1.

Claim 10
The method as described in claim 1 wherein the processing results in one or more program instructions adapted to be performed by the first processor, the
method further comprising: writing the program instructions to the common memory; notifying the first processor that the program instructions have been written; and executing the program instructions by the first processor. As per claim 1.

(Note Applicant Examiner elected to not scan in the changes to the claims in this section. The amendments was entered and the mapping of the rejection is taught by the claims above. Examiner tried to avoid a duplication of work).

Claim 11
An information handling system comprising: a plurality of heterogeneous processors; a common memory shared by the plurality of heterogeneous processors; a first processor selected from the plurality of processors that sends a code processing request to a second processor, the second processor also being selected from the plurality of processors; a local memory corresponding to the second processor; a DMA controller associated with the second processor, the DMA controller adapted to transfer data between the common memory and the second processor's local memory; and a processing tool for processing software code, the processing tool including software effective to: receive, at a second processor, the code processing request requested by the first processor; write software code data corresponding to the request to the second processor's local memory in response to the request; and process the software code data by the second processor. As per claims 1and 7.

Claim 12
The information handling system as described in claim 11 further comprising software code effective to: prior to the reception of the request: read script code from the common memory; write the script code to a local memory corresponding to the first processor; interpret, at the first processor, the script code, the interpreting resulting in the software code; and
write the software code to the second processor's local memory. As per claim 2.

Claim 13
The information handling system as described in claim 11 further comprising software code effective to: write data resulting from the executing to the common memory. As per claim 3.

Claim 14
The information handling system as described in claim 11 further comprising software code effective to: prior to the reception of the request:
run a first program, during the running of the first program, identify a call to the software code; and load the software code into the common memory, wherein the processing of the software code is occurs simultaneously to the running of the first program and wherein the processing of the software code is completed prior to the call of the software code from the first program.

As per claim 4.

Claim 15
The information handling system as described in claim 14 further comprising software code effective to: perform a multimedia effect resulting from the processing of the software code, the performance performed by the second processor. As per claim 5.

Claim 16
The information handling system as described in claim 14 further comprising software code effective to: receive, at the first processor, executable instructions resulting from the processing performed by the second processor, wherein the executable instructions are adapted to perform a multimedia effect; and perform the multimedia effect on the first processor by executing the received executable instructions. As per claim 6.

Claim 17
The information handling system as described in claim 11 further comprising software code effective to:
load, a the second processor, a virtual machine program into the second processor's local memory; read, from the common memory shared by the first and second processors, the software code data that includes virtual machine code adapted to be processed by the virtual machine program; process the virtual machine code at the second processor using the virtual machine program, the processing resulting in executable instructions; write, using a DMA operation, the executable instructions to a memory location accessible by the first processor; and execute, at the first processor, the executable instructions. As per claim 7.

Claim 18
The information handling system as described in claim 17 wherein the memory location is selected from the group consisting of a local memory corresponding to the first processor, and the common memory. As per claim 8.

Claim 19
The information handling system as described in claim 17 wherein the first and second processors are dislike processors with different instruction set architectures and wherein the executable instructions are adapted to be executed on the first processor and not the second processor. As per claim 9.

Claim 20
The information handling system as described in claim 11 wherein the process results in one or more program instructions adapted to be performed by the first processor, the information handling system further comprising software code effective to: write the program instructions to the common memory; notify the first processor that the program instructions have been written; and execute the program instructions by the first processor. As per claim 10.

Claim 21

A computer program product stored on a computer operable media for processing software code, said computer program product comprising:
means for receiving, at a second processor, a code processing request requested by a first processor, wherein the first and second processors are heterogeneous processors within a computer system that share a common memory;
means for writing software code data corresponding to the request to a local memory corresponding to the second processor in response to the request; and
means for processing the software code data by the second processor. As per claim 1.

Claim 22
The computer program product as described in claim 21 further comprising:
prior to the means for receiving:
means for reading script code from the common
memory; means for writing the script code to a local memory corresponding to the first processor; means for interpreting, at the first processor, the script code, the interpreting resulting in the software code; and means for writing the software code to the second processor's local memory. As per claim 2.

Claim 23
The computer program product as described in claim 21 further comprising:
means for writing data resulting from the executing to the common memory.
As per claim 3.

Claim 24
The computer program product as described in claim 21 further comprising:
prior to the means for receiving:
means for running a first program, during the running of the first program, identifying a call to the software code; and
means for loading the software code into the common memory, wherein the processing of the software code is occurs simultaneously to the running of the first program and wherein the processing is completed prior to the call of the software code from the first program.
As per claim 4.

Claim 25
The computer program product as described in claim 24 further comprising:
means for performing a multimedia effect resulting from the processing of the software code, the performance performed by the second processor. As per claim 5.

Claim 26
The computer program product as described in claim 24 further comprising:
means for receiving, at the first processor, executable instructions resulting from the processing performed by the second processor, wherein the executable instructions are adapted to perform a multimedia effect: and means for performing the multimedia effect on the first processor by executing the received executable instructions. As per claim 6.

Claim 27
The computer program product as described in claim 21 further comprising:
means for loading, a the second processor, a virtual machine program into the second processor's local memory; means for reading, from the common memory shared by the first and second processors, the software code data that includes virtual machine code adapted to be processed by the virtual machine program; means for processing the virtual machine code at the second processor using the virtual machine program, the processing resulting in executable instructions; means for writing the executable instructions to a memory location accessible by the first processor using a DMA operation; and means for executing, at the first processor, the executable instructions. As per claim 7.

Claim 28
The computer program product as described in claim 27 wherein the memory location is selected from the group consisting of a-local memory corresponding to the first processor, and the common memory. As per claim 8.

Claim 29
The computer program product as described in claim 27 wherein the first and second processors are dislike processors with different instruction set
architectures and wherein the executable instructions are adapted to be executed on the first processor and not the second processor. As per claim 9.

Claim 30
The computer program product as described in claim 21 wherein the means for processing results in one or more program instructions adapted to be performed by the first processor, the computer program product further comprising:
means for writing the program instructions to the common memory;
means for notifying the first processor that the program instructions have been written; and
means for executing the program instructions by the first processor.
As per claim 10.

Conclusion

7. Applicant's response clarified many of the issues. The Examiner has made this action non final.

Correspondence Information

8. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Todd Ingberg whose telephone number is (571) 272-3723. The examiner can normally be reached on during the work week..

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng-Ai An can be reached on (571) 272-3756. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Todd Ingberg
Primary Examiner
Art Unit 2193

TI